H0005001 US

CONFIGURABLE LEVELS OF SOURCE CONTROL FOR THE CONFIGURATION OF A PROCESS AUTOMATION SYSTEM

BACKGROUND OF THE INVENTION

1. Field of the Invention

[0001] The present disclosure generally relates to source control and process automation systems. In particular, the present disclosure relates to version control, regulated industry, development, qualification, process automation, and other applications and features.

2. Discussion of the Background Art

[0002] Source control is also known as configuration management, version control, and change management. Source control is a discipline of making changes to source code in a planned and systematic fashion. The purpose of source control is to formally control the integrity of artifacts (items) and activities (tasks). In a source control system, objects are checked-out, edited, and then checked-in. Each time an object is checked-in, it is given a version number. Over time, a history of changes is created for the objects under the control of the source control system.

[0003] The items under control in a source control system include objects, such as control strategies. In object-oriented programming (OOP), objects are abstractions used in designing a program and they are also the units of code that are eventually derived from the design process. In between, each object is made into a generic class of objects and even more generic classes are defined so that objects can share models and reuse the class definitions in their code. Each

object is an instance of a particular class or subclass with the class' own methods or procedures and data variables. Thus, objects typically exist in a hierarchy of objects with parent and child relationships. An object is usually a binary, text, or other type of file.

[0004] Process control systems are also known as process automation systems and are used to control and monitor complex processes in many types of industrial settings, including refineries, pharmaceuticals, power and chemical plants, and pulp, paper and printing mills. One example is Experion PKS™ available from Honeywell.

[0005] Different users of process automation systems have different needs in regard to source control of the application configuration in the control system. On one end of the spectrum are users that do not need to track any changes or versions. On the other end are users, often in regulated industries, that need very tight control over the change and version management of development projects.

[0006] There is a need for different levels of source control in a process control system that are capable of being enabled according to user need, such as manual source control, basic source control, and full source control.

SUMMARY OF THE INVENTION

[0007] The present invention has many aspects and is directed to configurable levels of version control for the configuration of a process automation system that fulfills the above needs and more.

[0008] One aspect is a method of source control. A level of source control is enabled from a selection of at least two levels. A version number of an object is set either automatically or manually, depending on the level. In some embodiments, the method includes providing a capability to switch the level of source control to another level. In some embodiments, automatically setting the version number is based on the degree of change to the object. In some embodiments, the method includes storing attributes associated with the object in a database. In some embodiments, the object is a control strategy loadable to a controller in a process control system.

[0009] Another aspect is a process control system, comprising a computer, a network, and a controller. The computer has a source control system with a selectable level of source control. The network couples the computer to the controller. At least one control strategy in the source control system is loadable from the computer to the controller. In some embodiments, the system includes a database to store source control information associated with the control strategy, including a version number. In some embodiments, the selectable level of source control is no source control and a version number is entered manually when the control strategy is saved. In some embodiments, the selectable level of source control is basic source control and a version number is automatically incremented when the control strategy is saved. In some embodiments, the selectable level of source control is full source control and a version number is automatically incremented when the control strategy is checked-in. In some embodiments, the selectable level of source control is a preference, a license, an installation configuration, or a user interface.

[0010] Another aspect is a method for providing a source control system for a process control system. A selection from at least two levels of source control is received. A user-enterable version number is provided when an object

is stored in the source control system, if the selection is a first level. An automatically incremented version number is provided when an object is stored, if the selection is a second level. In some embodiments, an automatically incremented version number is provided when the object is checked-in, if the selection is a third level. In some embodiments, the selection is changed to another of the levels of source control. In some embodiments, attributes of the object are updated based on the selection.

[0011] A technical solution and technical effect is providing for levels of source control in a process control system where the levels are capable of being enabled according to user need. Another technical solution and technical effect is providing an option of three specific levels, manual source control, basic source control, or full source control. Thus, there is increased flexibility and adaptability.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] These and other features, aspects, and advantages of the present disclosure will become better understood with reference to the following description, appended claims, and drawings where:

[0013] FIG. 1 is a screenshot of an example user interface for a source control system having a first level of source control;

[0014] FIG. 2 is a screenshot of an example user interface for a source control system having a second level of source control;

[0015] FIG. 3 is a screenshot of an example user interface for a source control system having a third level of source control;

[0016] FIG. 4 is a flow chart of an example method for providing configurable levels of support for a source control system;

[0017] FIG. 5 is a flow chart of an example method for changing configurable levels of support for a source control system;

[0018] FIG. 6A and 6B together form a flow chart of an example method for updating version attributes of an object based on the level of source control;

[0019] FIG. 7 is a block diagram of an example system architecture for operating a source control system in a process control system; and

[0020] FIG. 8 is a block diagram of another example system architecture for operating a source control system in a process control system.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0021] FIGS. 1, 2, and 3 show an example source control system with three levels of source control. The present invention is not limited to a selection of three levels. A range of any number of selectable levels of source control ranging from a low level of control to a high level of control are used in some embodiments. In this example, a first level of source control is called manual or "none", a second level of source control is called "basic", and a third level of source control is called "full". The selection of the level of source control may be performed in any manner, such as through a license, registration, installation, configuration, user interface or other selection method or combination of methods.

[0022] In this example system, there are three levels of source control from low to high: level one is manual source control ("none"); level two is "basic" source control; and level three is "full" source control. With manual source control, the user manually maintains a version number as part of each control object. With "basic" source control, source control is enabled through a preference menu and version numbers are automatically incremented when a

control object changes. With "full" source control, the version number is not only incremented but a repository of all versions of each control object is maintained. Additional features for full source control include security to prevent unauthorized changes, a full audit trail of interactions per control object, and enforcement of a user-defined development life cycle. The life cycle may vary from a very simple life cycle to an extensive and detailed life cycle depending on the industry. With all three levels in this example system, the system maintains parameters related to the creation, revision date, and user.

[0023] In this example system, automatic increments of a version number for basic source control are determined by the system. An increment is either a minor increment, such as +0.01 for minor changes or a major increment, such as +1 for major changes. Examples of major changes are addition or deletion of a function block and addition or deletion of a parameter connection. A function block is a control object characterized by a set of parameters and an algorithm. A parameter connection occurs when one object defines a parameter that is used in another object. In this example, the version number is shown when an object is displayed or a report is printed.

[0024] In this example system, other operations can also result in changes to a version number such as copying, and importing. With copying, the new object's version number is zero and the revision date is the current date. With importing, the object takes the version number and date from an import file.

[0025] FIG. 1 is a screenshot of an example user interface for a source control system having a first level of source control. FIG. 1 shows a main user interface for an input/output (I/O) module 100. The I/O module is one example of a number of objects under control of the source control system in this example. Other objects in this example include controller objects, fieldbus interface

modules (FIMs), control execution environments, and container blocks. The present invention is not limited to these types of objects. The objects in the source control system are manually assigned version numbers by a user in this first level of source control. In FIG. 1, there is an entry space 102 for the user to enter a version number for this I/O module. The characteristics of this example system with "none" or manual source control, include user-entered text for a version, a created-by name set by a first save changes operation, a modified-by name set by a save changes operation, a date-created date set by a first save changes operation, a version date set by a save changes operation, comments that are user-entered, and no version history. A save changes operation is the processing of a request by a user to the source control system to store changes to an object.

[0026] FIG. 2 is a screenshot of an example user interface for a source control system having a second level of source control. FIG. 2 shows a project view user interface 200 displaying information for a project named "pidloop". The project has a version number associated with it and displayed next to the project name 202. The characteristics of this example system with "basic" source control include automatically generated version numbers, version numbers incremented upon saving changes, (e.g., +0.01 for minor changes and +1 for major changes), a created-by name set by a first save changes operation, a modified-by name set by a save changes operation, a date-created date set by a first save changes operation, a version date set by a save changes operation, comments entered by a user, no version history, and a version number displayed in trees, chart titles, and other places.

[0027] FIG. 3 is a screenshot of an example user interface for a source control system having a third level of source control. FIG. 3 shows information for an object named "CM_1_34_INST" 300 having a version number 302. The characteristics of this example system with "full" source control include a

qualification life cycle support, a version control system toolbar and menu item, automatically generated version numbers, version numbers incremented on check-in , (e.g., +0.01 for minor changes and +1 for major changes determined by the user), version number shown in trees and chart titles, a created-by name set by a first check-in to the source control system, a modified-by name set by a check-in, a date-created date set by a check-in, a date created set by a first check-in, a version date set by a check-in, comments entered by a user, and a full version history and audit trail.

[0028] A summary of the features associated with the three levels of source control in this example system are shown in Table 1. VERSIONNUM is a parameter associated with an object containing a version number. QUALSTATE is a parameter associated with an object containing a qualification state or life cycle phase. BLCKCOMMENT1...BLCKCOMMENT4 are parameters associated with an object containing a comment associated with an object.

Parameter or Feature	None	Basic	Full
Enabling mechanism	None	Set via a system preference	Set via a license
Display of version number in titles	None	VERSIONNUM is used as source of displayed number	VERSIONNUM is used as source of displayed number
Version menu and toolbar	No	No	Yes
VERSION	User entered text	Formatted string of VERSIONNUM	Formatted string of VERSIONNUM
VERSIONNUM	Not used	Auto-incremented by save changes	Incremented when checked in to the source control system, major/minor set by user
CREATEDBY	Set by first save changes	Set by first save changes	Set when first checked in
MODIFIEDBY	Set by save changes	Set by save changes	Set when checked in
DATECREATED	Set by first save changes	Set by first save changes	Set when first checked in

VERSIONDATE	Set by save changes	Set by save changes	Set when checked in
QUALSTATE	Not used	Not used	Set by user (only when already checked in)
BLCKCOMMENT1...BLCKCOMMENT4	User entered	User entered	User entered

Table 1. Summary of Version Parameter Behavior

[0029] FIG. 4 shows an example method for providing configurable levels of support for a source control system. In step 400, a request for a particular level of support is received by the source control system. In step 402, it is determined if full source control is licensed. In some embodiments, a level of source control is determined by a license. In other embodiments, a user interface provides the selection of level of source control. In still other embodiments, other methods of selecting a level of source control are used. If full source control is licensed, then in step 404, the source control level is set to "full" and control goes to step 406 where the level of source control is returned. Otherwise, if full source control is not licensed, then control flows to step 408. In step 408, it is determined if the option for basic source control is set. If so, control flows to step 410, where the source control level is set to "basic" and control goes to step 406 where the source control level is returned. Otherwise, if the option for basic source control is not set, control flows from step 408 to step 412. In step 412, the source control level is set to "none" and control goes to step 406, where the source control level is returned. In various embodiments, there are less than or more than three levels and the levels have various names or numbers.

[0030] FIG. 5 is a flow chart of an example method for changing configurable levels of support for a source control system. In step 500, a request to change the level of source control is received by the source control system. In step 502, it is determined if the user has the privilege to change the level of source control. If not, control flows to step 504, where a status is set to indicate that the

action cannot be performed. In this and other flow charts, error or status processing is shown, but it is optional. In various embodiments of the present invention, none or different methods of error or status processing is performed. From step 504, control flows to step 506, where the status is returned. Otherwise, if in step 502, it was determined that the user had the privilege, control flows to step 508.

[0031] In step 508, it is determined if "full" source control is licensed. If so, control flows to step 510, where the status is set to indicate that the action cannot be performed and, then, control flows to step 506 where the status is returned. Otherwise, if "full" source control is licensed, control flows to step 512. In step 512, it is determined if the change is from "none" to "basic". If so, control flows to step 514. In step 514, any needed processing is performed to change from "none" to "basic" and, then, control flows to step 516, where the level of support is updated in a database. Otherwise, if it was determined in step 512 that the change was not from "none" to "basic", then control flows to step 518. In step 518, it is determined if the change is from "basic" to "none." If so, any needed processing is performed to change from "basic" to "none" and, then, control flows to step 516, where the level of support is updated in the database and, then, to step 506, where the status is returned.

[0032] Some more detailed example methods of changing configurable levels of a source control system follow for migrating from a "none" or no source control system to a "basic" source control system, for migrating from a "basic" source control system to a "full" source control system, and for reverting to a lower level.

[0033] Migrating from a "none" or no source control system to a "basic" source control system occurs when a customer purchases a source control system

for an existing process control system without any source control in this example. Because there is no way to determine values of certain parameters, such as the original creator of an object, they are left null. However, the creation date is set to be the date of the migration and the version number for each object is set to 1.0.

[0034] Migrating from a "basic" source control system to a "full" source control system occurs when a customer purchases a license to upgrade to "full" in this example. The user checks in all of the objects. All of the "basic" parameters associated with each object are maintained by the source control system and are used when each object is checked in for the first time under the "full" system.

[0035] Reverting to a lower level of source control occurs when a customer does not renew a "full" license and reverts to "basic" or decides not to use its "basic" source control anymore and to work without source control ("none"). When reverting to "basic" or "none", the version numbers reset to zero. In "basic" the version numbers are then automatically generated, while in "none" they are manually entered from that point on.

[0036] FIG. 6A and 6B together form a flow chart of an example method for updating version attributes of an object based on the level of source control. Version attributes include information associated with an object, such as the date created, the version number and the like. In step 600, a save changes request for the object is made. In step 602, it is determined whether the object is a new object. If so, control flows to step 604, where a new version number is set to zero and, then, control flows to step 606, where the version attributes are updated. FIG. 6B describes the process of updating the version attributes in more detail. In step 602, if the object is not a new object, then control flows to step 608. In step 608, it is determined whether "full" source control is licensed. If so, control flows to step 610, where no actions are performed at this time and an update is

performed by the normal check-in processing and, then, control flows to step 612. In step 612, the method terminates after the save changes request is processed. In step 608, if "full" source control is not licensed, then control flows to step 614. In step 614, it is determined whether the source control level is "basic". If so, control flows to step 616, where the current version number is retrieved, the new version number is set to be an increment of the current version number and, then, control flows to steps 606 and 612.

[0037] FIG. 6B describes the process of updating the version attributes of step 606 in FIG. 6A in more detail. In step 618, it is determined whether "full" source control is licensed. If so, control flows to step 620, where the version attributes supported for "full" source control support are updated. Then, control flows to step 622, where the version attributes for "basic" source control support are updated. After step 622, control flows to step 624. In step 624, the version attributes for "none" source control support are updated. Then, control flows from step 624 to step 626, which is the end of version attribute updating. If in step 618, it is determined that "full" source control is not licensed, then control flows to step 628. In step 628, it is determined if the level of source control support is "basic". If so, control flows to step 622, otherwise control flows to step 624. In other words, for "full" source control support, all the attributes for "full", "basic", and "none" are updated; for "basic" source control support, all the attributes for "basic" and "none" are updated; and for "none" source control support, only the attributes for "none" are updated.

[0038] FIG. 7 shows an example of a system architecture for operating a source control system in a process control system. A client/server PC 700 has a source control system operating on it and is coupled via a network 702 to a controller 704 which communicates with various devices to provide process control. The client/server PC 700 acts as both a client and a server so that a user

has access to the source control system and the ability to command a load. Control strategies in a source control system are loaded from client/server 700 to controller 704.

[0039] FIG. 8 shows another example of a system architecture for operating a source control system in a process control system. Client PCs 800 are used by operators to control a plant. Client PCs 800 communicate with redundant server PCs 802. Redundant server PCs 802 are data engines or servers that provide data to and from the plant to client PCs 800. Redundant server PCs 802 are an alternative for greater availability than the single server PC 700 of FIG. 7. When redundant server PCs 802 are used, a back-up server takes over if a primary server fails. Control strategies in a source control system are loaded by PC 800 or server 802 to a controller 804 over a network 806. Controllers 804 communicate with various devices in a process control system.

[0040] It is to be understood that the above description is intended to be illustrative and not restrictive. Many other embodiments will be apparent to those of skill in the art upon reviewing the above description, such as adaptations of the present disclosure to source control systems for applications other than process control systems. Various designs using hardware, software, and firmware are contemplated by the present disclosure, even though some minor elements would need to change to better support the environments common to such systems and methods. The present disclosure has applicability to fields outside process control, such as software development environments and other kinds of systems needing control system configuration. Therefore, the scope of the present disclosure should be determined with reference to the appended claims, along with the full scope of equivalents to which such claims are entitled.